IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Before the Board of Patent Appeals and Interferences

In re the Application

Inventer	:	**Pinto et al.**
Application No.	:	**10/561,454**
Filed	:	**December 20, 2005**
For	:	**DATA PROCESSING DEVICE WITH INSTRUCTION CONTROLLED CLOCK SPEED**

APPEAL BRIEF

On Appeal from Group Art Unit 2183

Dan Piotrowski
Registration No. 42,079

Date: August 30, 2009

By: Thomas J. Onka
Attorney for Applicant
Registration No. 42,053

TABLE OF CONTENTS

I. REAL PARTY IN INTEREST

Koninklijke Philips Electronics N.V. is the real party in interest.

II. RELATED APPEALS AND INTERFERENCES

There are no related appeals or interferences.

III. STATUS OF CLAIMS

As filed, this case included claims 1-12. Claims 1-2, 4-8 and 10-12 remain pending, stand rejected, and form the basis of this appeal. Claims 3 and 9 have been cancelled without prejudice.

IV. STATUS OF AMENDMENTS

This appeal is in response to an Office Action, dated September 9, 2008 and a Final Office Action, dated March 30, 2009. Claims 1-2, 4-8, and 10-11 stand rejected under 35 USC 103(a) as being unpatentable over Sih et al. (USP 6,606,700, hereinafter Sih), Hennessey et al. ("Computer Organization and Design: the Hardware/Software Interface", hereinafter Hennessey), and Sager et al. (USP 6,487,675, hereinafter Sager), and claim 12 under 35 U.S.C. 103(a) over Sih, Hennessey, Sager, and Kim et al. (USPA 2004/02225868, hereinafter Kim).. On January 6, 2009, an amendment in response to the Office Action dated September 9, 2008, was entered by the Examiner. On May 26, 2009, in response to the Final Office Action dated March 30, 2009, an amendment was entered.

On June 8, 2009, an Advisory Action was entered into the record. The Advisory Action stated that the arguments filed on May 26, 2009 are not persuasive. In response, a Notice of Appeal was filed on June 30, 2009.

V. SUMMARY OF CLAIMED SUBJECT MATTER

The present invention, particularly, independent claim 1 discloses an instruction controlled data processing device comprising an instruction issue unit that is configured to issue respective ones of instructions of program code in successive instruction cycles, the instructions including at least a first type of instruction and a second type of instruction, a clocking circuit that is configured to clock the instruction cycles, a register file with a read port and a write port, a plurality of functional units, (See page 1, lines 19-29; FIG 1), each functional unit having a control input coupled to the issue unit, an operand input coupled to the read port and a result output coupled to the write port, and a control unit coupled to the issue unit, that is configured to route the result output of a first functional unit to the write port of the register file in response to instructions of the first type, and to the operand input of a second functional unit during an instruction cycle in response to instructions of the second type, wherein the clock circuit is configured to vary a rate of clocking the instruction cycles in dependence upon whether a current segment of the program code includes one or more instructions of the second type. (See page 2, lines 9-29; FIG 1; page 3, line 29 – page 4, line 4).

Independent claim 10 discloses a method of executing a processing task, comprising the steps of providing a plurality of functional units, issuing successive instructions at an instruction cycle rate, executing those of the instructions that are of a first type each with an individual one of the functional units during one instruction cycle, executing an instruction that is of a second type with a first and a second one of the functional units in series during one instruction cycle, routing a result of the first one of the functional units to an operand of the second one of the functional units in response to the instruction of the second type; and selecting the instruction cycle rate from at least a first and second rate, based on the type of instruction, the first rate being so slow that execution of instructions of the second type by a cascade of at least two of the functional units fits within an instruction cycle at the first rate, the second rate being so fast that only execution of instructions of the first type fits within the instruction cycle at the second rate, execution of instructions of the second type not fitting within one instruction cycle at the second rate. (See page 1, line 19 – page 2, line 29; FIG 1; page 3, line 29 – page 4, line 4).

Claims 2 and 4-8 depend from independent claim 1 and recite further aspects of the invention claimed.

Claims 11-12 depend from independent claim 10 and recite further aspects of the invention claimed.

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

The issue in the present mater is whether:

(1) Rejection of: Claims 1-2, 4-8, and 10-11 under 35 U.S.C. 103(a) over Sih et al. (USP 6,606,700, hereinafter Sih), Hennessey et al. ("Computer Organization and Design: the Hardware/Software Interface", hereinafter Hennessey), and Sager et al. (USP 6,487,675, hereinafter Sager), and claim 12 under 35 U.S.C. 103(a) over Sih, Hennessey, Sager, and Kim et al. (USPA 2004/02225868, hereinafter Kim) is in error.

VII. ARGUMENT

(1) Rejection of claims 1-12

Appellants respectfully submit that the rejection of claims 1-2, 4-8, and 10-11 under 35 U.S.C. 103(a) over Sih et al. (USP 6,606,700, hereinafter Sih), Hennessey et al. ("Computer Organization and Design: the Hardware/Software Interface", hereinafter Hennessey), and Sager et al. (USP 6,487,675, hereinafter Sager), and claim 12 under 35 U.S.C. 103(a) over Sih, Hennessey, Sager, and Kim et al. (USPA 2004/02225868, hereinafter Kim) is in error.

The present invention relates to a data processing device which has a plurality of

functional units and issues instructions in successive instruction cycles. Instructions of a first type are each intended for one functional unit at a time. An instruction of a second type causes a combination of functional units to respond in the same instruction execution cycle, a result from one functional unit being used by another as part of the execution of the same instruction. Preferably, the device supports alternative operation at a number of different instruction cycle rates, dependent on whether an executed program segment contains instructions of the second type. The fastest instruction cycle rate does not allow execution of the instruction of the second type, because operation by different functional units does not fit within the instruction execution cycle. When possible, the device saves power by switching to a slower clock rate, in which case instructions of the second type are executed to save additional power, by reducing the number of instructions that have to be issued.

It is respectfully submitted that in order to establish a *prima facie* case of obviousness, three basic criteria must be met;

> 1. there must be some suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to modify the reference or combine the reference teachings;
>
> 2. there must be a reasonable expectation of success; and
>
> 3. the prior art reference must teach or suggest all the claim limitations. The teaching or suggestion to make the claimed combination and the reasonable expectation of success must be found in the prior art, and not based on applicant's disclosure. *In re Vaeck*, 947 F.2d 488, 20 USPQ2d 1438 (Fed. Cir. 1991)

In KSR Int'l. Co. v. Teleflex, Inc., the Supreme Court noted that the analysis supporting a rejection under 35 U.S.C. 103(a) should be made explicit, and that it is "important to identify a reason that would have prompted a person of ordinary skill in the relevant field to combine the [prior art] elements" in the manner claimed:

> "Often, it will be necessary ... to look to interrelated teachings of multiple patents; the effects of demands known to the design community or present in the marketplace; and the background knowledge possessed by a person having ordinary skill in the art, all in order to determine whether there was an **apparent reason** to combine the known elements in the fashion claimed by the patent at issue. To facilitate review, this analysis **should be made explicit.**" KSR, 82 USPQ2d 1385 at 1396 (emphasis added).

Further, MPEP 2143 states:

> "If the proposed modification would render the prior art invention being modified unsatisfactory for its intended purpose, then there is no suggestion or motivation to make the proposed modification."

Sih teaches a dual-multiply-accumulate (dual-MAC) architecture that performs four multiply-shift-add operations in parallel. Sih's design is intended to provide high-speed computations typically required for real-time FIR filtering by providing multiple "single-cycle" multiply-accumulate (MAC) operations, and each of the parallel MACs must operate at substantially the same speed to achieve this single-cycle operation. The speed of operation will be dependent upon the speed of each of the multiply, shift, and add elements in each of the parallel MACs.

Sager teaches the use of multiple clock domains to allow latency-tolerant operations, such as fetch and decode operations, to be performed at a lower speed than latency-intolerant

functions, such as core arithmetic operations. If it is known, for example, that the processing of each data item is going to consume an amount of time, T, that is substantially longer than the time required to fetch each data item, there is no need to fetch each data item at a maximum fetch rate, and the clock of the fetch unit can be slowed to the slowest rate that still provides the data item every T time periods.

Of particular note, in Sager's architecture, the elements in the different clock domains are expected to be able to perform their tasks in parallel. The aforementioned fetch operation, for example, is expected to fetch the next data item while the previous data item is being processed. It is this parallelism that provides latency-tolerance; an outer clock domain can afford to spend as much time at a task as its inner clock domains will allow. Alternatively stated, the outer clock domain elements are not on the 'critical path' that determines the overall delay of the device. Without parallelism, all operations would be on the critical path and would need to be performed as quickly as possible, with no tolerance for unnecessary latency.

Sih's multiply, shift, and add elements within each parallel MAC cannot operate at sub-optimal clock rates and still provide their intended high-speed multiply-accumulate (MAC) function. They must each perform their respective function within one instruction cycle, and that cycle should not be lengthened by the inclusion of an unnecessary latency in any of these functions. Sih's multiply, shift, and add elements operate in series on the critical path, and the best achievable instruction cycle time is based on the serial addition of the time required by each element to perform its function. There is no opportunity in Sih's architecture for latency-tolerance among these functional elements. Without latency-tolerance, Sager's teachings cannot be applied.

The Office action asserts that one of skill in the art would apply Sager's teachings to Sih's architecture "for the advantage of decreased chip space usage and power savings". This assertion is incorrect. Sager achieves this decreased chip space usage and power savings by reducing the design constraints on the elements in the outer clock domains, off the critical path. Using smaller transistors will reduce chip space, but also increase transition time for driving a given load; a reduced clock rate will accommodate this increased transition time and consume less power. That is, the savings achieved by Sager are achieved at the cost of increased delay time, and this increased delay time is permitted because it is applied to the latency-tolerant elements that are off the critical path in the outer clock domains.

Given that the purpose of Sih's design is to provide high-speed MAC operations, one of skill in the art would optimize all of the elements along the critical path subject to a given set of design constraints. One of skill in the art would not be motivated to apply techniques that only provide an advantage for latency-tolerant elements to Sih's latency-intolerant multiply, shift, and add elements, as asserted by the Examiner. If a slower instruction cycle rate were acceptable, one of skill in the art would design all of Sih's multiply, shift, and add elements to operate at this slower instruction cycle rate, because it would then be the most efficient in area and power consumption for the given instruction cycle rate. Sager's degradation of speed for selective elements would be a sub-optimal design compared to a consistent degradation of speed for all elements in series along the critical path.

Because there is no apparent reason to apply Sager's teachings to the multiply, shift, add elements of Sih, as asserted by the Examiner, and because the application of Sager's teachings to these elements would be unsatisfactory for Sih's intended purpose, the applicants respectfully maintain that the rejection of claims 1-2, 4-8, and 10-11 under 35 U.S.C. 103(a) over Sih, Hennessey, and Sager is unfounded, and should be withdrawn.

Further, assuming, *in argument*, that a combination of Sih, Hennessey, and Sager were to be created, such a combination will not provide the elements of each of the applicants' independent claims 1 and 10.

The combination of Sih, Hennessey, and Sager fails to teach or suggest an instruction issue unit that issues instructions of program code in successive instruction cycles, the instructions including at least a first type of instruction and a second type of instruction and a plurality of functional units, each functional unit having a control input coupled to the issue unit, and fails to teach or suggest a clock circuit that varies a rate of clocking the instruction cycles in dependence upon whether a current segment of the program code includes one or more instructions of the second type, as explicitly claimed in claim 1, upon which claims 2 and 4-8 depend.

The Examiner asserts that Sih teaches instructions of two types "MAC and Dual-MAC instructions execute on processing paths MAC1 and MAC2" (Office action, page 4, lines 7-8). This assertion is incorrect. Sih does not teach "MAC and Dual-MAC" instructions, and the Examiner fails to identify where Sih provides this teaching.

Sih teaches a dual-MAC architecture, including a dual-MAC coprocessor, for a total of four MAC units. Sih does not teach that this architecture is configurable to distinguish between single and dual MAC instructions. Of particular note, Sih's architecture does not execute different instructions at all. Whenever Sih's circuit of FIG. 1 is activated, it will always perform the same set of operations. Each MAC will perform a 17x17 bit multiplication, a shift operation, and a 40-bit addition. The only programmable control over Sih's circuit is a selection of register inputs to each MAC, the number of bits to shift, the input to one of Sih's 40 bit adders, and the register outputs from each MAC. Sih's multiplication 104, 106, 128, 142 and addition 118, 120, 132, 146 elements are not programmable, and they will always perform their multiplication and addition functions whenever the MACs are activated. Sih does not teach that some or all of these elements are not activated so as to perform operations using different numbers of MACs, as asserted by the Examiner.

Additionally, the Examiner acknowledges that Sih and Hennessey fail to teach a clock circuit that is configured to vary a rate of clocking the instruction cycles in dependence upon the type of instruction being executed, and asserts that Sager provides this teaching. This assertion is incorrect. Sager teaches a different clock rate for different functional elements, but does not teach or suggest ***varying*** the instruction clock rate that is provided to any of the functional elements. Each of Sager's clock circuits 220, 225, 265, 270 provides a constant clock rate based on the master clock rate; none of these clock circuits are configured to ***vary*** their instruction clock rate based on the type of instruction being executed.

Because the combination of Sih, Hennessey, and Sager fails to teach or suggest the elements of claim 1, and because the Examiner's characterizations of the prior art are in error, the applicants respectfully maintain that the rejection of claims 1-2 and 4-8 under 35 U.S.C. 103(a) over Sih, Hennessey, and Sager is unfounded, and should be withdrawn.

The combination of Sih, Hennessey, and Sager also fails to teach or suggest executing instructions that are of a first type each with an individual one of the functional units during one instruction cycle, executing an instruction that is of a second type with a first and a second one of the functional units in series during one instruction cycle; fails to teach or suggest routing a result of the first one of the functional units to an operand of the second one of the functional units in response to the instruction of the second type; fails to teach or suggest selecting the instruction cycle rate from at least a first and second rate, based on the type of instruction; fails to teach or suggest the first rate being so slow that execution of instructions of the second type by a cascade of at least two of the functional units fits within an instruction cycle at the first rate; fails to teach the second rate being so fast that only execution of instructions of the first type fits within the instruction cycle at the second rate; and fails to teach or suggest execution of instructions of the second type not fitting within one instruction cycle at the second rate, as claimed in claim 10, upon which claims 11-12 depend.

The Examiner repeatedly asserts that Sih teaches MAC instructions and dual-MAC instructions, but fails to identify where Sih identifies such different instructions. The Examiner references column 3, lines 35-56 of Sih to support this assertion, but this cited text does not teach different MAC and dual-MAC instructions:

> "FIG. 1 is, as noted above, a block diagram of the new architecture. The core architecture contains a coupled dual-MAC structure composed of MAC units MAC1 and MAC2. MAC1 fetches its multiplier operands from output ports PO2 and PO3 of the register file. The output of the multiplier (104) is passed to a shifter (108) that can shift the result left by 0, 1, 2, or 3 bits. The output of the shifter (108) is passed to an adder (114) that takes its other input from a multiplexer, MUX1 (116), that has zero and the result of the shifted product from MAC2 as its inputs. The output of the adder (114) is passed into a 40-bit adder (118) than can add another 40-bit operand fetched from output port PO1 of the register file. The output of the 40-bit adder is stored into the register file via input port PI1. MAC2 fetches multiplier operands from register file output ports PO4 and PO5, multiplies them (106), and shifts (110) the result left by 0, 1, 2, or 3 bits. The shifter output is passed to a 40-bit adder (120) that can add an additional register file operand fetched from output port PO6. The shifter output is also sent to the multiplexer, MUX1 (116) that feeds the first adder (114) in MAC1. The output of the 40-bit adder (120) is stored into the register file via register file input port PI2." (Sih, column 3, lines 35-56.)

As is clearly evident, at the cited text, Sih teaches the sequence of operations of Sih's dual-MAC architecture, and nowhere in this cited text does Sih distinguish between different types of instruction, and nowhere in this cited text does Sih teach "MAC" instructions that are different from "dual-MAC" instructions, as asserted by the Examiner. Sih's MAC elements respond to a single 'execute' command, the different functions of FIGs. 2, 4, and 5 being provided by controlling which registers 100 are connected to each MAC input and output, and controlling the multiplexers 112, 116, 124, 126, 136, 140, 150 that route the signals within the MACs.

The Examiner notes that "Dual-MAC instructions are executed in series by multiplication followed by addition". The applicants note that in Sih, ***all*** instructions are executed by multiplication followed by addition; there is no way in Sih not to perform a multiplication followed by addition, other than to perform no operation at all.

The Examiner acknowledges that Sih and Hennessey fail to teach selecting the instruction cycle rate from at least a first and second rate, based on the type of instruction, the first rate being so slow that execution of instructions of the second type by a cascade of at least two of the functional units fits within an instruction cycle at the first rate, the second rate being so fast that only execution of instructions of the first type fits within the instruction cycle at the second rate, execution of instructions of the second type not fitting within one instruction cycle at the second rate, as claimed in claim 10, and asserts that Sager provides this teaching at column 4, line 48 through column 5, line 6. This assertion is incorrect.

At the cited text, Sager teaches:

> "FIG. 3 illustrates the high-speed sub-core 205 of the processor 200 of the present invention. The high-speed sub-core includes the most latency-intolerant portions of the particular architecture and/or microarchitecture employed by the processor. For example, in an Intel Architecture processor, certain arithmetic and logic functions, as well as data cache access, may be the most unforgiving of execution latency.
>
> "Other functions, which are not so sensitive to execution latency, may be contained within a more latency-tolerant execution core 210. For example, in an Intel Architecture processor, execution of infrequently-executed instructions, such as transcendentals, may be relegated to the slower part of the core.
>
> "The processor 200 communicates with the rest of the system (not shown) via the I/O ring 215. If the I/O ring operates at a different clock frequency than the latency-tolerant execution core, the processor may include a clock mult/div unit 220 which provides clock division or multiplication according to any suitable manner and conventional means. Because the latency-intolerant execution sub-core 205 operates at a higher frequency than the rest of the latency-tolerant execution core 210, there may be a mechanism 225 for providing a different clock frequency to the latency-intolerant execution sub-core 205. In one mode, this is a clock mult/div unit 225." (Sager, column 4, line 48 - column 5, line 6.)

As is clearly evident, the cited text teaches applying different clock rates to different functional elements; it does not teach an issue unit that issues instructions at one of two different rates, a first rate that is so slow that execution of instructions of the second type by a cascade of at least two of the functional units fits within an instruction cycle at the first rate and a second rate being so fast that only execution of instructions of the first type fits within the instruction cycle at the second rate, as claimed in claim 10.

Because the combination of Sih, Hennessey, and Sager fails to teach the elements of claim 10, the applicants respectfully maintain that the rejection of claims 10-11 under 35 U.S.C. 103(a) over Sih, Hennessey, and Sager is unfounded, and should be reversed.

With regard to the dependent claim 12, this claim is dependent upon claim 10, and in this rejection, the Examiner relies on the combination of Sih, Hennessey, and Sager for teaching the elements of claim 10. As noted above, there is no apparent reason to combine Sager and Sih as proposed by the Examiner, and even if such a combination were formed, the combination of Sih, Hennessey, and Sager fails to teach the elements of claim 10, and Kim fails to correct these deficiencies. Accordingly, the applicants respectfully maintain that the rejection of claim 12 under 35 U.S.C. 103(a) over Sih, Hennessey, Sager, and Kim that relies on the combination of Sih, Hennessey, and Sager for teaching the elements of claim 10 is unfounded, and should be withdrawn.

VIII. CONCLUSION

In view of the above analysis, it is respectfully submitted that the referred to reference fails to anticipate the subject matter of any of the present claims. Therefore, reversal of all outstanding grounds of rejection is respectfully solicited.

Respectfully submitted,

Dan Piotrowski
Registration No. 42,079

Date: August 30, 2009

By: Thomas J. Onka
Attorney for Applicant
Registration No. 42,053

IX. CLAIMS APPENDIX

1. An instruction controlled data processing device comprising:

an instruction issue unit that is configured to issue respective ones of instructions of program code in successive instruction cycles, the instructions including at least a first type of instruction and a second type of instruction;

a clocking circuit that is configured to clock the instruction cycles;

a register file with a read port and a write port;

a plurality of functional units, each functional unit having a control input coupled to the issue unit, an operand input coupled to the read port and a result output coupled to the write port; and

a control unit coupled to the issue unit, that is configured to route the result output of a first functional unit to the write port of the register file in response to instructions of the first type, and to the operand input of a second functional unit during an instruction cycle in response to instructions of the second type;

wherein the clock circuit is configured to vary a rate of clocking the instruction cycles in dependence upon whether a current segment of the program code includes one or more instructions of the second type.

2. The processing device of claim 1, organized as a VLIW processor, wherein the issue unit includes a plurality of issue slots for issuing a VLIW instruction word, the register file having a plurality of sets of read and write ports, the functional units or groups of functional units each coupled to a respective one of the issue slots and the sets of read and write ports.

3. (Canceled)

4. The processing device of claim 1, wherein the clock circuit includes a plurality of selectable clock rates, including a first clock rate that is sufficiently slow to accommodate a latency of instructions of the second type involved in producing a result from the second functional unit during execution of the instruction of the second type within the instruction cycle, and a second clock rate that is too fast to accommodate the latency of instructions of the second type in the instruction cycle, but accommodates latency of instructions of the first type.

5. The processing device of claim 1, wherein the control unit is configured to selectively route the result output of a third functional unit to a further operand input of the second functional unit under control of the instruction of the second type.

6. The processing device of claim 5, wherein the program code includes a VLIW instruction that contains a command for the third functional unit and the instruction of the second type for issue in a same instruction cycle.

7. The processing device of claim 1, wherein the control unit is arranged to make the second functional unit respond to the instruction of the second type in an instruction execution cycle following an instruction execution cycle in which the first functional unit responds to the instruction of the second type.

8. The processing device of claim 7, wherein the result of the first functional unit is routed without intermediate latching from the first functional unit to the operand input of the second functional unit.

9. (Canceled)

10. A method of executing a processing task, comprising:

providing a plurality of functional units,

issuing successive instructions at an instruction cycle rate;

executing those of the instructions that are of a first type each with an individual one of the functional units during one instruction cycle,

executing an instruction that is of a second type with a first and a second one of the functional units in series during one instruction cycle;

routing a result of the first one of the functional units to an operand of the second one of the functional units in response to the instruction of the second type; and

selecting the instruction cycle rate from at least a first and second rate, based on the type of instruction, the first rate being so slow that execution of instructions of the second type by a cascade of at least two of the functional units fits within an instruction cycle at the first rate, the second rate being so fast that only execution of instructions of the first type fits within the instruction cycle at the second rate, execution of instructions of the second type not fitting within one instruction cycle at the second rate.

11. The method of claim 10, including

issuing the successive instructions each as part of a VLIW instruction word that contains a plurality of instructions for respective further functional units;

including in the instruction word that contains the instruction of the second type a further instruction for a particular one of the further functional units; and

routing a further result of the further instruction from the particular one of the further functional units to a further operand input of the second one of the functional units in response to the instruction of the second type.

12. The method of claim 10, including adapting the instructions used to execute the processing task to the selected instruction cycle rate, so that the instructions of the second type are used when the task is executed at the first rate and the instructions of the second type are replaced by instructions of the first type with corresponding effect when the task is executed at the second rate.

X. EVIDENCE APPENDIX

No evidence has been submitted.

XI. RELATED PROCEEDINGS APPENDIX

There are no related proceedings.